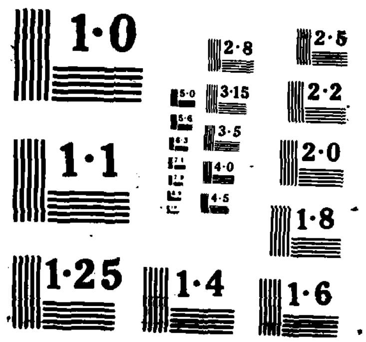
AD-A176 022 THIRTY-TWO NODES HEXAHEDRONAL ELEMENT SUBROUTINE FOR MULTI-PURPOSE PROGRAM NEF(U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA A SUKANEEYOUTH SEP 86 1/1

UNCLASSIFIED F/G 9/2 NL


1·0
2·8
2·5
5·0
3·15
2·2
5·6
3·5
1·1
4·0
2·0
4·5
1·8
1·25
1·4
1·6

②

AD-A176 022

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

THIRTY-TWO NODES HEXAHEDRONAL ELEMENT
SUBROUTINE FOR MULTI-PURPOSE PROGRAM MEF

by

Anan Sukaneeyouth

September 1986

Thesis Advisor: Gilles Cantin

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.
2b DECLASSIFICATION / DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S)	5 MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Naval Postgraduate School 6b OFFICE SYMBOL (If applicable) 69	7a. NAME OF MONITORING ORGANIZATION Naval Postgraduate School
6c. ADDRESS (City, State, and ZIP Code) Monterey, CA 93943-5000	7b. ADDRESS (City, State, and ZIP Code) Monterey, CA 93943-5000
8a NAME OF FUNDING/SPONSORING ORGANIZATION 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code)	10 SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO / PROJECT NO / TASK NO / WORK UNIT ACCESSION NO

11 TITLE (Include Security Classification)
THIRTY-TWO NODES HEXAHEDRONAL ELEMENT SUBROUTINE FOR MULTI-PURPOSE PROGRAM MEF

12 PERSONAL AUTHOR(S)
Anan Sukaneeyouth, LT, Thailand Navy

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
Engineers Thesis	FROM ____ TO ____	1986 September	39

16 SUPPLEMENTARY NOTATION

17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Consistent Load Vector, Computer Program, Finite Element, Integration Points, Jacobian Matrix, Reference Element, Shape Function.

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

A general finite element program of moderate complexity called MEF is organized to contain a library of one, two, and three-dimensional elements for the solution of problems from a wide variety of diciplines. A cubic, thirty-two node, three dimensional isoparametric element was developed. With such an element very complex structures could be solved with a very course mesh.

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21 ABSTRACT SECURITY CLASSIFICATION Unclassified	
22a NAME OF RESPONSIBLE INDIVIDUAL Professor Gilles Cantin	22b TELEPHONE (Include Area Code) (408) 646-2364	22c OFFICE SYMBOL 69Ci

DD FORM 1473, 84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE

Approved for public release; distribution is unlimited.

Thirty-Two Nodes Hexahedronal Element Subroutine
for Multi-Purpose Program MEF

by

Anan Sukaneeyouth
Lieutenant, Royal Thai Navy
B.S., Royal Thai Naval Academy, 1978

Submitted in partial fulfillment of the
requirements for the degrees of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING
and
MECHANICAL ENGINEER

from the

NAVAL POSTGRADUATE SCHOOL
September 1986

Author: Anan Sukaneeyouth
Anan Sukaneeyouth

Approved by: Gilles Cantin
Gilles Cantin, Thesis Advisor

Paul ... for
A.J. Healey, Chairman,
Department of Mechanical Engineering

J.N. Dyer
John N. Dyer,
Dean of Science and Engineering

ABSTRACT

A general finite element program of moderate complexity called MEF is organized to contain a library of one, two, and three-dimensional elements for the solution of problems from a wide variety of diciplines. A cubic, thirty-two node, three dimensional isoparametric element was developed. With such an element very complex structures could be solved with a very course mesh.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

I. INTRODUCTION

A. A GENERAL-PURPOSE FINITE ELEMENT PROGRAM

A general-purpose finite element program should be able to solve a variety of problems from the number of disciplines: linear and non-linear, static and dynamic problem of elasticity, fluid mechanics, heat transfer, etc. and can solve problems of large size involving a variety of elements.

A general program is going to be voluminous and complex. It is, however, desirable that:

- its logic be easily understood;
- one or many of its parts be easily modifiable;
- it offers possibilities to tailor its facilities for the solution particular classes of problems.

The program should have a modular structure, with the modules made as independent from one another as is practicable. The following modular operations are recognized:

1. **Problem Definition (data base):**
   - node coordinates and element connectivities;
   - nodal element properties;
   - boundary conditions.
2. **Element Computations:**
   - integration points and associated weights;
   - interpolation functions and their derivatives;
   - Jacobian matrix, inverse, and determinant;
   - element matrices and vectors: [k], [m], {f}, etc.
3. **Assembly Operations:**
   - assemblage of master matrices and vectors, [K], [M], etc.
4. **Solution:**
   - factorization of master matrices and solution of equations.
5. **Result:**
   - output of nodal variables and other calculated quantities: gradients, reactions, etc.

Subroutines implementing the various operations described above are contained in all finite element codes. The flow of information between these operations is problem dependent; linear, non-linear, static, and dynamic problems all require logic of their own.

B. MEF PROGRAM

A program of medium complexity, called MEF, implementing the techniques of general-purpose program that can solve a large variety of boundary value problems of mathematical physics. It is written in FORTRAN IV and can be easily adaptable to various computers.

The main program controls the flow of all information through the functional blocks by transferring control to a subroutine called BLNNNN when the block calling card NNNN is encountered in the input file. The subroutine BLNNNN then performs preliminary functions such as logical unit identification, and reading of control parameters for the creation of various files and tables. The subroutine then calls subroutine EXNNNN. In all cases, subroutine BLNNNN provides appropriate default parameters which will be overridden by user values if specified. Subroutine EXNNNN then performs the major operations of the block by calling on the needed subroutines in the MEF library. The above protocol holds for all blocks except STOP, COMT, and IMAG. All the functions of COMT and IMAG are performed by subroutine BLNNNN, and the function of block STOP is performed by the main program.

The executable functional blocks contained in the MEF are:

BLOCK	FUNCTION
SOLR	Assemblage of distributed load.
LINM	Solution of linear problem with global matrix in core.
LIND	Solution of linear problem with global matrix out of core.
NLIN	Solution of stationary non-linear problem.
TEMP	Solution of unsteady problem (linear or non-linear).
VALP	Eigenvalues and eigenvectors.

The various blocks designed for execution of the various computations have similar structures since they have to:

- construct element and load matrices;
- assemble global matrices and vectors;
- factorize and solve the system of equations;

- output the results.

Using the constructed elements and load matrices, the subroutine element library ELEMLB is called. This library contains subroutines that define the individual element types. The ELEMO3 subroutine, a thirty-two node, three dimensional isoparametic element was developed and added to the element library. This new element allowed the solution of linear elastic structures composed of homogeneous and isotropic materials.

Multiple sample problems were developed to fully exercise use of this new element. An indepth investigation was then conducted to determine the limit of computational ability using the newly defined element to represent physical phenomenons.

II. DESCRIPTION OF THE COMPUTER PROGRAM

A. REFERENCE ELEMENT

To simplify the analytical expression for elements of complex shapes an element of reference is introduced. Such an element is defined in an abstract non-dimensional space with a very simple geometrical shape. The geometry of the reference element is then mapped into the geometry of the real element using geometrical transformation expression.

A thirty-two node, three dimensional cubic element was introduced as the reference element. The element has eight coner nodes and twenty four mid-side nodes dividing each edge in three equal parts as shown in Figure 2.1. Using this reference element we created the fundamental matrices and vectors in subroutine called ELEM03, NI03, D03 and B03 to be used in element library subroutine ELEMLB of the MEF program.

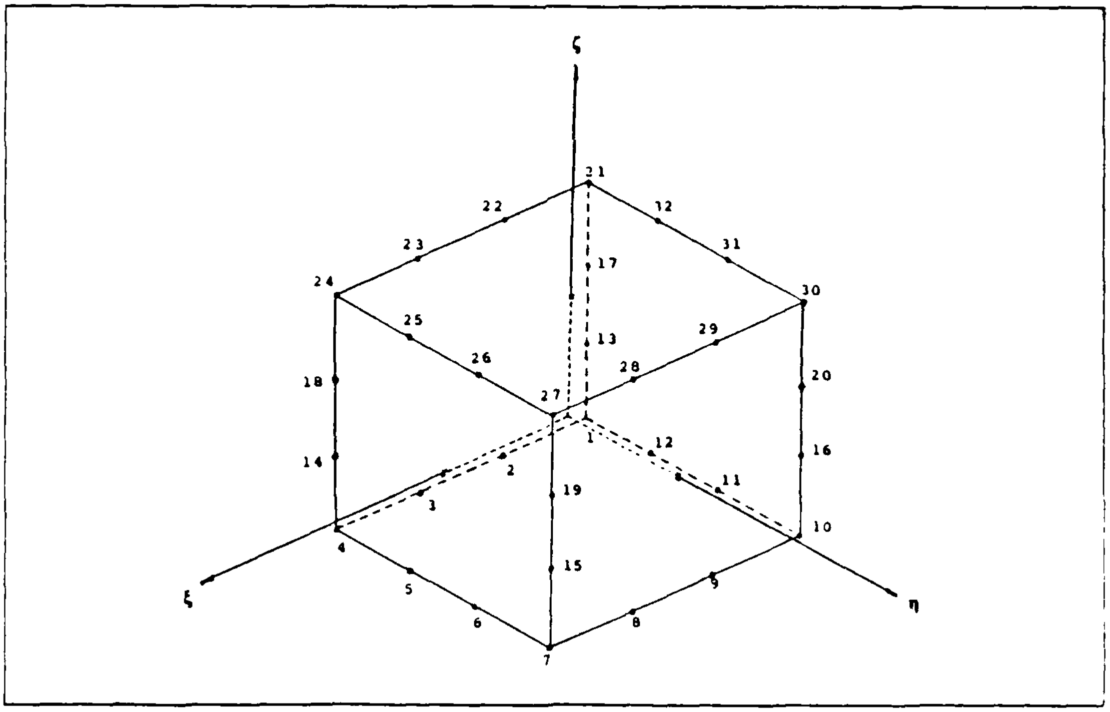


Figure 2.1 Reference Element.

B. SUBROUTINE ELEM03

Multi-purpose program MEF is organized to contain a library of one, two, and three-dimensional elements for the solution of problems from a wide variety of disciplines. Problems from the mechanics of solids and fluids, heat transfer, etc. have been solved. For each element type ***nn***, subroutine ELEMnn controls the computation of all matrices and vectors.

In element type 03, subroutine ELEM03 is organized to create the fundamental matrices and vectors that must be numerically integrated using methods of Numerical Integration described in and the computations can be carried out by the following steps.

1. **Operation common to all element of the same type:**
   - compute the weight w_r and the coordinate of integration points;
   - construct the functions N (interpolation functions), the function $\overline{N}$ (geometrical interpolation functions) and their derivatives with respect to ξ, η, ζ at the points of integration.
2. **Operation for the computation of matrix [k] of each element:**
   - initialize the matrix [k];
   - for each point of integration ξ_r :
     - construct the Jacobian matrix [J] from the derivatives with respect to ξ, η, ζ of function N and the nodal coordinates of the element;
     - construct the inverse of [J] and its determinant;
     - construct the derivatives of functions N with respect to x, y, z starting from the derivatives with respect of ξ, η, ζ ;
     - construct the matrices [B] and [D];
     - accumulate into [k] the values of $[B]^t[D][B]\det(J)w_r$ calculated for each integration point.
3. **Operation required to compute mass matrix [m]:**
   - initialize [m]
   - for each integration point ξ_r :
     - compute Jacobian matrix and its determinant;
     - accumulate the values of $\{N\} < N > \det(J)w_r$ into matrix [m].
4. **Operation required to compute consistent load vectors {f}:**
   - initialize {f};
   - for each integration point ξ_r :
     - compute the Jacobian matrix and its determinant;

- accumulate the values of $\{N\}f_V\ \det(J)w_r$ into $\{f\}$.

5. **Operation required to compute the residue vecter {r}:**
   - using the value of $\{f\}$ from 4;
   - for each integration point ξ_r :
     - compute matrices [B], [D], [J] as in 2 above;
     - accumulate the product: $\{f\} - [B]^t[D][B]\{u_n\}\ w_r\det(J)$ into $\{r\}$.
6. **Operation required to compute gradients ∂u at points of integral:**
   - for each integration point ξ_r :
     - construct matrix [B] as in 2 above;
     - compute and print gradient $\{\partial u\} = [B]\{u_n\}$.

The subroutine ELEM03 executes one operation at a time depending on the value of ICODE. Control variable ICODE specifies which element operation is desired, and expression of this variable as follows:

ICODE = 1 initialization of the characteristic parameters of an element (number of nodes, number of degrees of liberty).

ICODE = 2 operations required by a given referance element which are independent of the real geometry; construction of interpolation function N and their derivative with respect to ξ at the points of integration.

ICODE = 3 construction of matrix [k] in array VKE.

ICODE = 4 construction of tangent matrix needed for non-linear problems in array VKE.

ICODE = 5 construction of mass matrix [m] in array VKE.

ICODE = 6 computation of residual vector {r} in array VFE.

ICODE = 7 computation load vector {f} in array VFE.

ICODE = 8 computation and printing of gradients $\{\partial u\}$.

C. CODING.

1. **Evaluate coordinates, weights, functions N and their derivatives**

Integration formular for numerical integration is the following form.

$$I = \sum_{i=1}^{ipg} w_l y(\xi_l) \qquad \text{(eqn 2.1)}$$

where

- ξ_l are the coordinates of integration point 'l' in ξ, η, ζ system coordinate corresponding to weight w_l ;

- w_l are the weights corresponding to integration point number;
- IPG are the total number of integration points.

A choice of 2, 3 or 4 integration points by dimension can be made in subroutine GAUSS [Ref. 1:p. 265], giving respectively 8, 27 and 64 integration points, the weights corresponding to the integration points and their coordinates. They are in the array called IPG, VCPG and VKPG respectively.

Subroutine NI03 create the array VNI that contains the shape function N_i and their derivatives with respect to ξ_i (ξ, η, ζ system coordinate) as Figure 2.2.

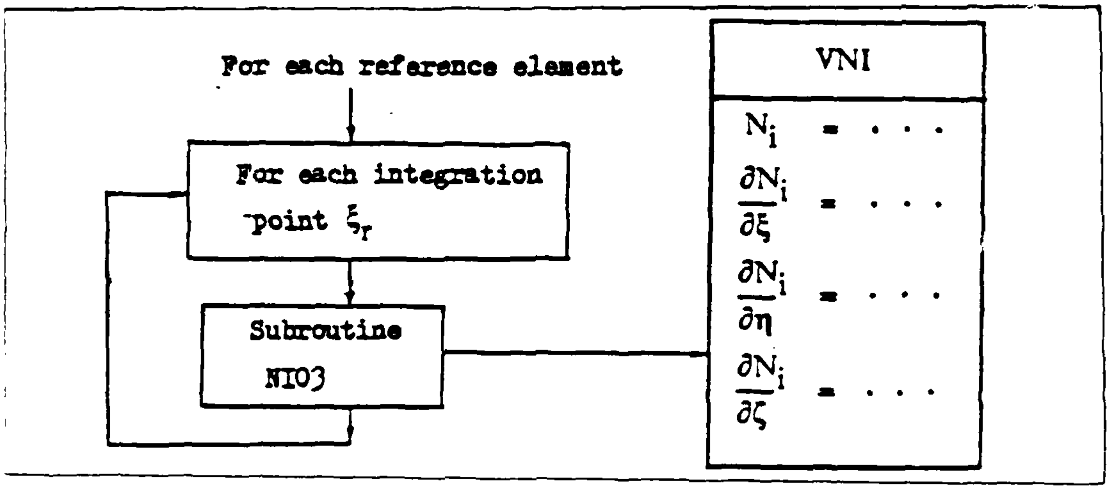


Figure 2.2 Block Diagram for the Shape Function and their Derivative.

2. **Computation of stiffness matrix [k] of each element.**

The explicit equation of element stiffness matrix is following:

$$[k] = \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} [B]^t[D][B]\det(J)d\zeta\, d\eta\, d\xi \qquad \text{(eqn 2.2)}$$

where

- [B] is the linear strain matrix;
- $[B]^t$ is the transpose matrix of [B];
- [D] is the matrix of elastic constants for an isotropic material.

The stiffness matrix that has the block diagram as Figure 2.3 is the integration of product of the linear strain transpose matrix, the element property matrix and the linear strain matrix over the volume of the reference element.

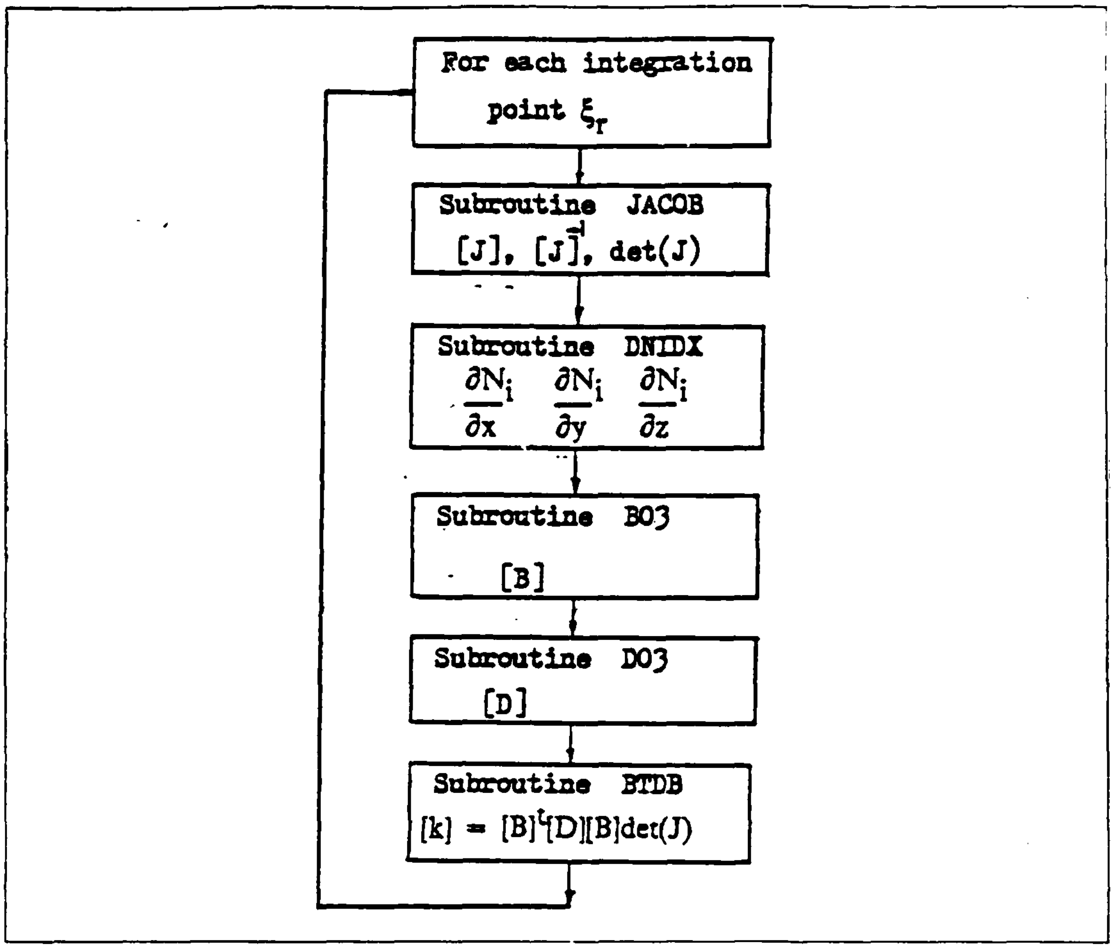


Figure 2.3 Block Digram of Computation of Stiffness Matrix.

In coding, we employ numerical integration formular having the following generic form:

$$[k] = \sum_{i=1}^{ipg} w_i [k^o(\xi_i)] \qquad \text{(eqn 2.3)}$$

where

- IPG are the total number of integration points;
- w_i are the weighting coefficients corresponding to each integration point;
- ξ_i are the coordinate of the IPG integration points;
- $[k^o]$ is the stiffness matrix at each integration point as shown in equation 2.4 .

$$[k^o] = [B]^t[D][B]\det(J) \qquad \text{(eqn 2.4)}$$

Subroutine JACOB [Ref. 1:p. 63], compute the Jacobian matrix, its inverse and its determinant. The Jacobian matrix as Figure 2.4 is obtained as the product of two matrices, one containing the derivatives of the geometrical transformation functions with respect to the space of the reference element, and the other containing the real coordinates of the geometrical nodes of the element.

$$< x\ y\ z > \; = \; < N(\xi) > [\{x_n\}\ \{y_n\}\ \{z_n\}] \qquad \text{(eqn 2.5)}$$

$\{x_n\}$, $\{y_n\}$, $\{z_n\}$ being the geometrical node coordinates. The Jacobian matrix is figure 2.4

$$[J] = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} & \frac{\partial z}{\partial \eta} \\ \frac{\partial x}{\partial \zeta} & \frac{\partial y}{\partial \zeta} & \frac{\partial z}{\partial \zeta} \end{bmatrix}$$

Figure 2.4 The Jacobian Matrix.

The inverse of Jacobian matrix $[J]^{-1}$ as Figure 2.5 and its determinant are computed by the method discribed on [Ref. 1:p. 44],

Subroutine DNIDX [Ref. 1:p. 64], computes the derivatives of the shape function N_i with respect to the coordinate system of the real element using the product on Figure 2.6 .

Subroutine B03 creates the matrix as Figure 2.7 that contains the strain components at all direction of each node in the element using output components of subroutine DNIDX and rearrange the new array called VBE.

Subroutine D03 compute the stress-strain matrix (VDE) as Figure 2.8 [Ref. 2:p. 110] for isotropic materials (E = Young's Modulus, ν = Pisson's Ratio).

$$[J]^{-1} = \begin{bmatrix} \frac{\partial\xi}{\partial x} & \frac{\partial\eta}{\partial x} & \frac{\partial\zeta}{\partial x} \\ \frac{\partial\xi}{\partial y} & \frac{\partial\eta}{\partial y} & \frac{\partial\zeta}{\partial y} \\ \frac{\partial\xi}{\partial z} & \frac{\partial\eta}{\partial z} & \frac{\partial\zeta}{\partial z} \end{bmatrix}$$

Figure 2.5 The Inverse of Jacobian Matrix.

$$\begin{Bmatrix} \frac{\partial N_i}{\partial x} \\ \frac{\partial N_i}{\partial y} \\ \frac{\partial N_i}{\partial z} \end{Bmatrix} = \begin{bmatrix} \frac{\partial\xi}{\partial x} & \frac{\partial\eta}{\partial x} & \frac{\partial\zeta}{\partial x} \\ \frac{\partial\xi}{\partial y} & \frac{\partial\eta}{\partial y} & \frac{\partial\zeta}{\partial y} \\ \frac{\partial\xi}{\partial z} & \frac{\partial\eta}{\partial z} & \frac{\partial\zeta}{\partial z} \end{bmatrix} \begin{Bmatrix} \frac{\partial N_i}{\partial \xi} \\ \frac{\partial N_i}{\partial \eta} \\ \frac{\partial N_i}{\partial \zeta} \end{Bmatrix}$$

Figure 2.6 The derivative of shape function w.r.t. x, y, z.

Subroutine BTDB construct the stiffness matrix by adding the product of the transpose matrix VBE, the stress-strain matrix VDE and the matrix VBE of every integration points.

$$[k] = [B]^t[D][B]w_i\det(J) \qquad \text{(eqn 2.6)}$$

3. **Computation The Mass Matrix [m].**

The element mass matrix has the block diagram as Figure 2.9 and the explicit equation as following:

$$[m] = \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} [N]^t[N]\det(J)\, d\zeta\, d\eta\, d\zeta \qquad \text{(eqn 2.7)}$$

and numerical integration form is:

$$
[B] = \begin{bmatrix}
N_{1,x} & 0 & 0 & N_{2,x} & 0 & 0 & \cdots\cdots & N_{32,x} & 0 & 0 \\
0 & N_{1,y} & 0 & 0 & N_{2,y} & 0 & \cdots\cdots & 0 & N_{32,y} & 0 \\
0 & 0 & N_{1,z} & 0 & 0 & N_{2,z} & \cdots\cdots & 0 & 0 & N_{32,z} \\
N_{1,y} & N_{1,x} & 0 & N_{2,y} & N_{2,x} & 0 & \cdots\cdots & N_{32,y} & N_{32,x} & 0 \\
0 & N_{1,z} & N_{1,y} & 0 & N_{2,z} & N_{2,y} & \cdots\cdots & 0 & N_{32,z} & N_{32,y} \\
N_{1,z} & 0 & N_{1,x} & N_{2,z} & 0 & N_{2,x} & \cdots\cdots & N_{32,z} & 0 & N_{32,x}
\end{bmatrix}
$$

Figure 2.7 The Array VBE.

$$
[D] = \frac{E(1-\nu)}{(1-\nu)(1-2\nu)} \begin{bmatrix}
1 & \frac{\nu}{1-\nu} & \frac{\nu}{1-\nu} & 0 & 0 & 0 \\
\frac{\nu}{1-\nu} & 1 & \frac{\nu}{1-\nu} & 0 & 0 & 0 \\
\frac{\nu}{1-\nu} & \frac{\nu}{1-\nu} & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & \frac{1-2\upsilon}{2(1-\upsilon)} & 0 & 0 \\
0 & 0 & 0 & 0 & \frac{1-2\upsilon}{2(1-\upsilon)} & \\
0 & 0 & 0 & 0 & 0 & \frac{1-2\upsilon}{2(1-\upsilon)}
\end{bmatrix}
$$

Figure 2.8 The Array VDE.

$$
[m] = \sum_{i=1}^{ipg} w_i [N]^t [N] \det(J) \qquad \text{(eqn 2.8)}
$$

- [N] is the shape function matrix Figure 2.10 that was created by subroutine NI03;
- $[N]^t$ is the transpose matrix of [N];
- w_i are the weighting coefficients corresponding to each integration point;

Obviously look at the product of matrices $[N]^t$and[N] as Figure 2.11 is a symmetric matrix. By convention, the mass matrix [m] contains upper half components and diagonal components of this matrix.

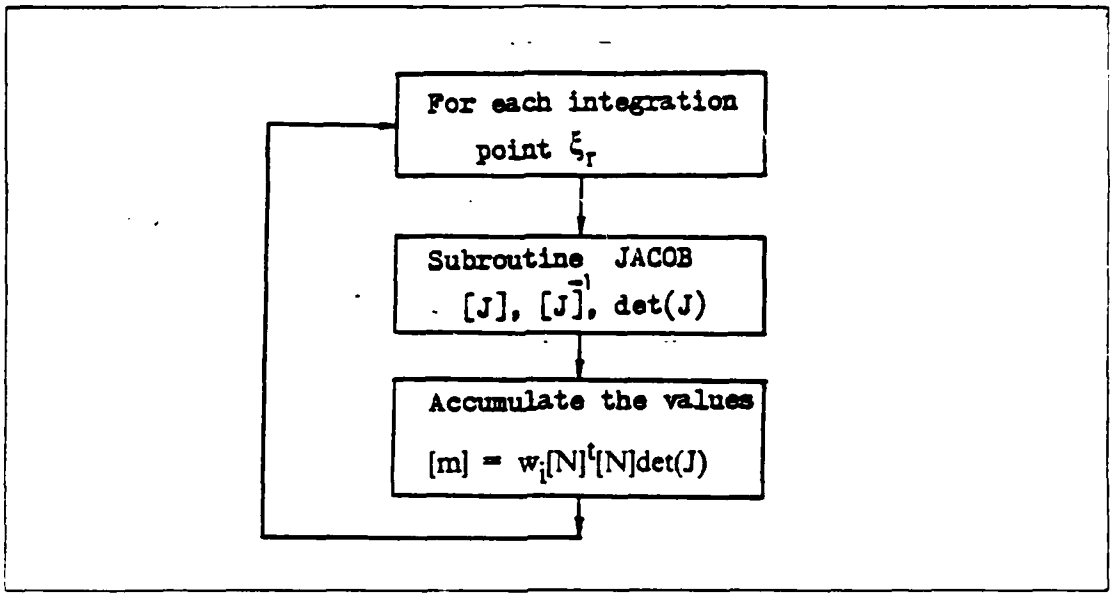


Figure 2.9 Block Diagram for the Mass Matrix [m].

$$[N] = \begin{bmatrix} N_1 & 0 & 0 & N_2 & 0 & 0 & \cdots & N_{32} & 0 & 0 \\ 0 & N_1 & 0 & 0 & N_2 & 0 & \cdots & 0 & N_{32} & 0 \\ 0 & 0 & N_1 & 0 & 0 & N_2 & \cdots & 0 & 0 & N_{32} \end{bmatrix}$$

Figure 2.10 The Shape Function Matrix [N].

4. **Computation of consistent load vector {f}.**

The explicit formular for the load vector {f} is:

$$\{f\} = \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}[N]^t\begin{Bmatrix} f_{vx} \\ f_{vy} \\ f_{vz} \end{Bmatrix}\det(J)\,d\zeta\,d\eta\,d\xi \qquad \text{(eqn 2.9)}$$

has the block diagram as Figure 2.12 and numerical integration form is:

$$\{f\} = \sum_{i=1}^{ipg} w_i[N]^t\begin{Bmatrix} f_{vx} \\ f_{vy} \\ f_{vz} \end{Bmatrix}\det(J) \qquad \text{(eqn 2.10)}$$

$$
[N]^t[N] = \begin{bmatrix}
N_1N_1 & 0 & 0 & N_1N_2 & 0 & 0 & N_1N_3 & 0 & 0 & \cdots \\
0 & N_1N_1 & 0 & 0 & N_1N_2 & 0 & 0 & N_1N_3 & 0 & \cdots \\
0 & 0 & N_1N_1 & 0 & 0 & N_1N_2 & 0 & 0 & N_1N_3 & \cdots \\
N_1N_2 & 0 & 0 & N_2N_2 & 0 & 0 & N_2N_3 & 0 & 0 & \cdots \\
0 & N_1N_2 & 0 & 0 & N_2N_2 & 0 & 0 & N_2N_3 & 0 & \cdots \\
0 & 0 & N_1N_2 & 0 & 0 & N_2N_2 & 0 & 0 & N_2N_3 & \cdots \\
N_1N_3 & 0 & 0 & N_2N_3 & 0 & 0 & N_3N_3 & 0 & 0 & \cdots \\
0 & N_1N_3 & 0 & 0 & N_2N_3 & 0 & 0 & N_3N_3 & 0 & \cdots \\
0 & 0 & N_1N_3 & 0 & 0 & N_2N_3 & 0 & 0 & N_3N_3 & \cdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots
\end{bmatrix}
$$

Figure 2.11 The Product of Matrices $[N]^t$ and [N].

f_{vx}, f_{vy}, f_{vz} are the force per unit volume in direction x, y, z.

5. **Computation the residue array.**

The element residual array has the block diagram as Figure 2.13 and defined by:

$$\{r\} = \{f\} - [k]\{u_n\} \qquad \text{(eqn 2.11)}$$

where

- $\{r\}$ is the residue vector;
- $\{f\}$ is the element load vector;
- $[k]$ is the element stiffness matrix;
- $\{u_n\}$ is the nodal values vector.

6. **Computation of the gradients and stresses at the points of integration.**

For each integration point:

- compute strain as Figure 2.14. [Ref. 3:p. 31]
  or compacted form

$$\{\varepsilon\} = [B]\{u_i\} \qquad \text{(eqn 2.12)}$$

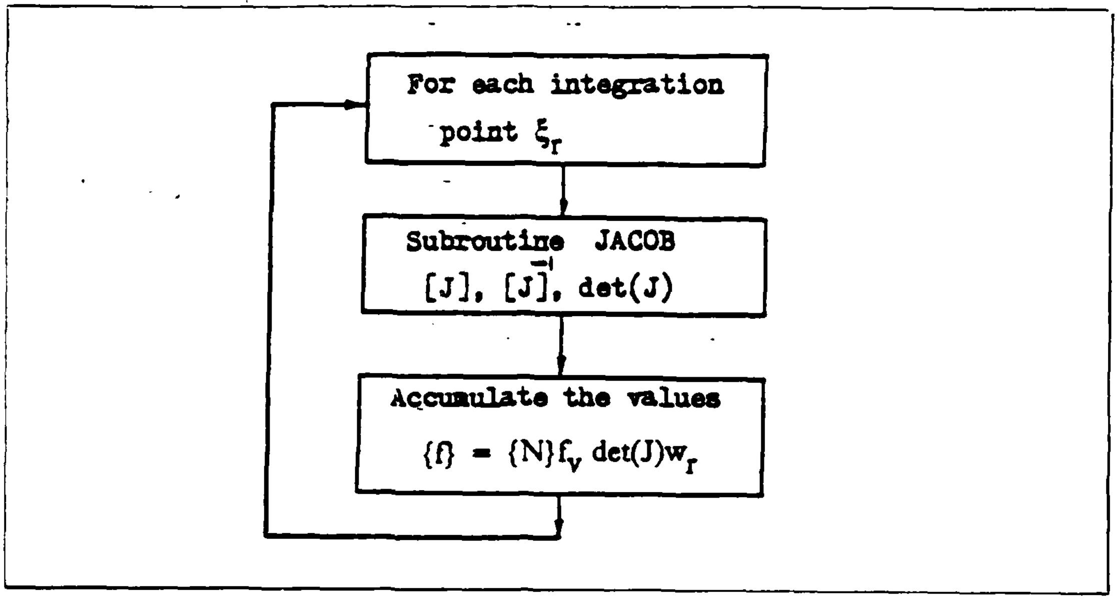


Figure 2.12 Block Diagram for Consistent Load Vector [f].

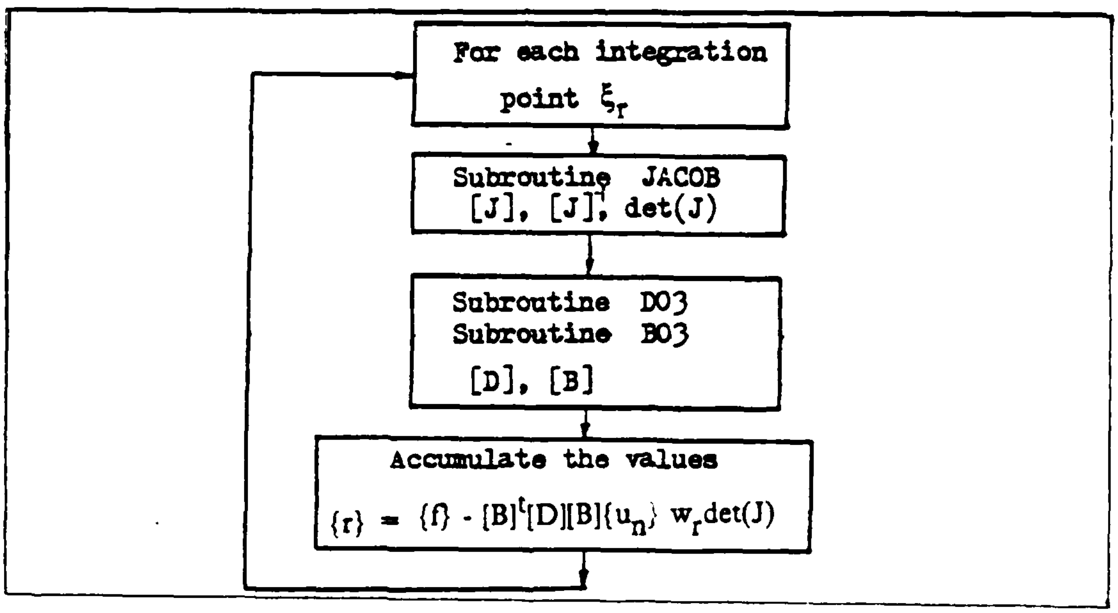


Figure 2.13 Block Diagram for the Residue Array.

- compute stress as Figure 2.15. [Ref. 3:p. 30]
  or compacted form

$$
\begin{Bmatrix} \varepsilon_x \\ \varepsilon_y \\ \varepsilon_z \\ \gamma_{xy} \\ \gamma_{yz} \\ \gamma_{zx} \end{Bmatrix}
=
\begin{bmatrix}
N_{1,x} & 0 & 0 & \cdots\cdots & N_{32,x} & 0 & 0 \\
0 & N_{1,y} & 0 & \cdots\cdots & 0 & N_{32,y} & 0 \\
0 & 0 & N_{1,z} & \cdots\cdots & 0 & 0 & N_{32,z} \\
N_{1,y} & N_{1,x} & 0 & \cdots\cdots & N_{32,y} & N_{32,x} & 0 \\
0 & N_{1,z} & N_{1,y} & \cdots\cdots & 0 & N_{32,z} & N_{32,y} \\
N_{1,z} & 0 & N_{1,x} & \cdots\cdots & N_{32,z} & 0 & N_{32,x}
\end{bmatrix}
\begin{Bmatrix} u_1 \\ v_1 \\ w_1 \\ \vdots \\ u_{32} \\ v_{32} \\ w_{32} \end{Bmatrix}
$$

Figure 2.14 The Strain-Node Value Relation.

$$
\begin{Bmatrix} \sigma_x \\ \sigma_y \\ \sigma_z \\ \tau_{xy} \\ \tau_{yz} \\ \tau_{zx} \end{Bmatrix}
=
\frac{E(1-\nu)}{(1-\nu)(1-2\nu)}
\begin{bmatrix}
1 & \frac{\nu}{1-\nu} & \frac{\nu}{1-\nu} & 0 & 0 & 0 \\
\frac{\nu}{1-\nu} & 1 & \frac{\nu}{1-\nu} & 0 & 0 & 0 \\
\frac{\nu}{1-\nu} & \frac{\nu}{1-\nu} & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & \frac{1-2\nu}{2(1-\nu)} & 0 & 0 \\
0 & 0 & 0 & 0 & \frac{1-2\nu}{2(1-\nu)} & 0 \\
0 & 0 & 0 & 0 & 0 & \frac{1-2\nu}{2(1-\nu)}
\end{bmatrix}
\begin{Bmatrix} \varepsilon_x \\ \varepsilon_y \\ \varepsilon_z \\ \gamma_{xy} \\ \gamma_{yz} \\ \gamma_{zx} \end{Bmatrix}
$$

Figure 2.15 The Stress-Strain Relation.

$$\{\sigma_i\} = [D]\{\varepsilon_i\} \qquad \text{(eqn 2.13)}$$

- compute coordinate of each integration points on the real element as figure 2.16.

  or compacted form

$$\{x_i\} = [N]\{x_{ni}\} \qquad \text{(eqn 2.14)}$$

Using x_n, y_n, z_n ,where $N = N(\xi_i, \eta_i, \zeta_i)$ and ξ_i, η_i, ζ_i are the gauss points, the coordinate of thirty two nodes on the real element that existed in the array VCORE . All the integration points are evaluated in the reference element. Using the product of the transfer matrix (***the shape function matrix N***) and the array VCORE, we can evaluate the coordinate of the integration points on the real element.

$$
\begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix} = \begin{bmatrix} N_1 & 0 & 0 & \cdots & N_{32} & 0 & 0 \\ 0 & N_1 & 0 & \cdots & 0 & N_{32} & 0 \\ 0 & 0 & N_1 & \cdots & 0 & 0 & N_{32} \end{bmatrix} \begin{Bmatrix} X_n \\ Y_n \\ Z_n \end{Bmatrix}
$$

Figure 2.16 Evaluation of the Coordinate of the Integration Points on the Real Element.

Note that subroutine ELEM03 executes one operation at a time depending on the value of ICODE. For preserving the memory locations, some of the operations have been performed using the same array name as VKE in both the stiffness matrix [k] and the mass matrix [m]. The same thing has been done with array VFE used for the residual vector {r} and the load vector {f}.

III. THE SOLUTION OF A SIMPLE PROBLEM AND DISCUSSION OF RESULT

In this chapter the preparation of input data for the computer program are described. A simple problem was developed and the investigation was conducted to determine the ability of this cubic element.

A. ENTRY AND EXECUTION FUNCTIONAL BLOCKS

MEF has specialized functional blocks for the entry, verification and organization of the data required to define a problem. Block COOR reads the nodal coordinates and the number of degrees of freedom of each node, it also provides automatic node generation. Block COND reads the boundary condition. Block PREL reads element properties if required for element type being used. Block SOLC reads the concentrated loads. Block ELEM reads the element connectivities; it also reads element group information when more than one element type is used, if elements have different properties. This block provides automatic element generation.

Other function blocks of MEF for the execution of particular finite element computations use the data base constructed by entry blocks and augment it by their results. Block LINM assembles and solves a linear system of equations residing in-core. Block LIND is similar to the block LINM but the system of equations resides out-of-core in a mass storage device. And block STOP terminates execution of the problem.

MEF provides various levels of output. The quantity of output desired from a given block is controlled by a parameter on the block calling card, described in detial [Ref. 1:pp. 440-447], which ranges from 0 (the assumed value) to 4. The default value provides all the information needed to verify the input stream and obtain the desired answers while the value 1 thru 4 provide various level of verbosity.

B. STRUCTURE MODELING

The first step in applying a finite element solution to the problem is the selection of an appropriate mesh. In many case an extrapolation of the results will be required and hence more than one mesh will have to be selected. The mesh size solution does not follow any predetermined rules and will, in general, depend on the nature of the problem and the judgement of the analyst. An abitrary numbering convention is adopted for the nodal points of the entire structure (in distinction with the numbering convention for the element nodes shown in Figure 3.1. A second numbering scheme is also needed for the elements of a mesh.

In order to show the function of the program, as well as some observations affecting the use of it, the solution of a simple problem is presented in this section. The problem selected is that usually presented in classical texts of strength of materials as a cantilever beam of uniform cross section subjected to a concentrated load at the end of the beam. The model used for this problem is shown in Figure 3.1. Nodes 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 and 12 (here arbitrarily numbered) are constrained to zero displacement in all directions. The concentrated load at the end of the cantilever beam is considered as the consistent load and are shown in Figure 3.2.

Four different meshes which each mesh has the thickness 9, 0.9, 0.09, 0.009 inches and has the concentrated loads 1200, 1.2, 1.2e-3 and 1.2e-6 pounds were employed in order to show the variation of results with mesh size. Numerical results for the maximum displacement at the tip of the beam are shown in Tables I and II.

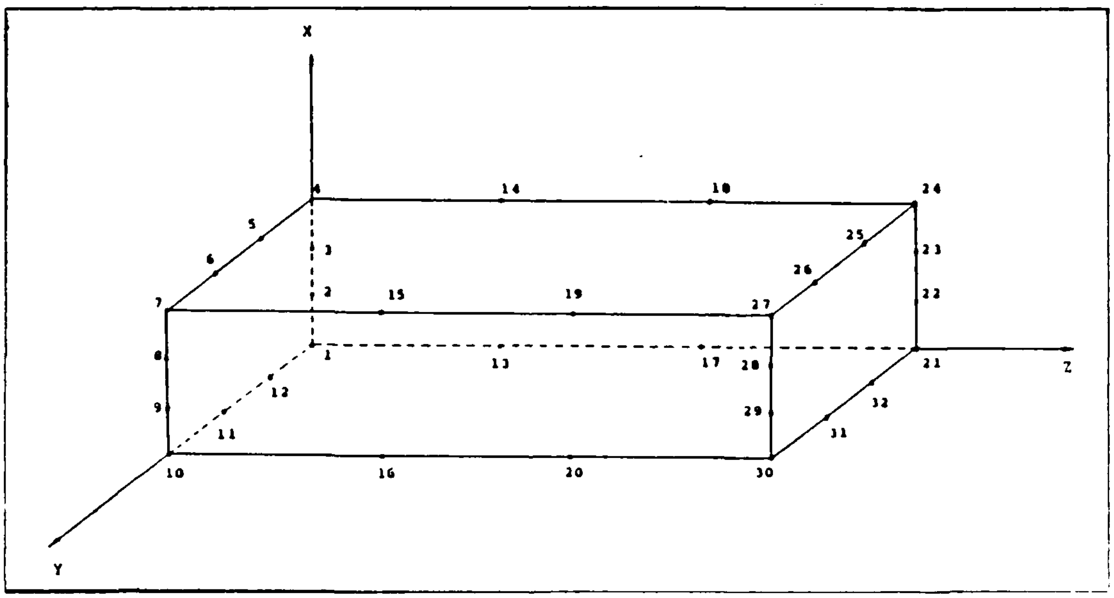


Figure 3.1 The Model used for the Problem.

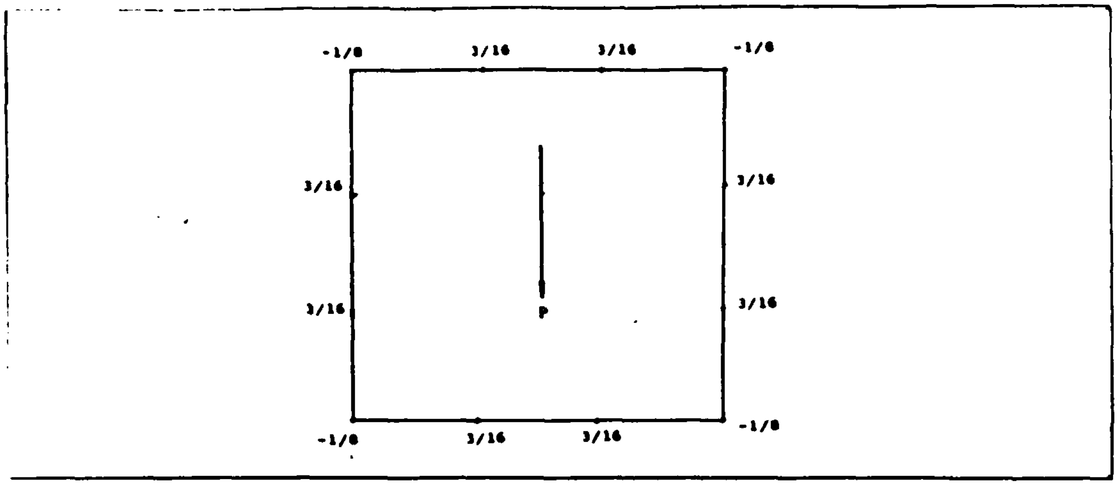


Figure 3.2 The consistent load at the end of the beam.

C. DISCUSSION OF RESULTS

Elementary beam theory gives a displacement of 0.0631 inches, however this beam theory neglects shear deflection and three dimensional elasticity does not do so. Observation of the results presented here reveal that mesh refinement lead to improved results which eventually will converge to a certain value. The thirty two nodal points brick is subject to numerical bad conditionary when used with an adverse slenderness ratio. This ratio being defined as the ratio of the maximum element dimension over the minimum dimension. For example in the numerical examples treated the slenderness ratio for the one and eight elements representation of the four beams analysis are shown in the Tables I and II. Results are acceptable for the first three beams in each case. However a computation of approximate values of zero, first invariant of the element stiffness as well as condition number are produced to investigate the results.

As the slenderness ratio increases the numerical of the conditioning of the stiffness matrices become so bad that no significant digits can be expected out of the solution for displacements. The same conclusion is arrived with a mesh of eight elements. In such a case the slenderness ratio varies from 1.6 to 1666.6. For the thickness of 0.09 the slenderness ratio was adequate. To reduce the ratio, more element must be used and tests with 20 and 40 elements were conducted. The twenty elements mesh gives of 0.6 to 666.6. In the use of a mesh with fourty elements the largest dimension become 6.0 and must therefore be reduced to 3.0 as 3.0 is the value of 120/40. Runs with such a mesh were then performed to confirm our results. On the basis of all these runs and the value of the condition number of these matrices it seems that a slenderness ratio of approximately 150 could still be used to give results with 6 significant digits in the answers, however further examples must be treated before a conclusion could be reached.

These are observations essentially made on the results of a simple problem. Thus no firm rules regarding the use of the newly element can be established and further experimention with different problems has to be conducted for this purpose.

TABLE I
THE DISPLACEMENT FOR THE ONE ELEMENT

Thickness	Slenderness Ratio	Zero	$\sum a_{ij}$	$\sum \lambda_i$	λ_{max}	λ_{min}	Condition Number	Displacement
9	13.3	0.24e-5	-0.54e-5	7.27e11	3.72e10	3.14e+4	1.18e06	-0.0581
0.9	133.3	0.76e-5	-0.16e-4	7.11e12	1.64e11	7.42e+1	2.22e09	-0.0515
0.09	1333.3	0.74e-4	0.16e-2	7.11e13	1.64e12	7.42e-2	2.22e13	-0.0486
0.009	13333.3	0.74e-3	0.56e-6	7.11e14	1.64e13	7.42e-3	4.59e15	*

TABLE II
THE DISPLACEMENT FOR THE EIGHT ELEMENTS

Thickness	Slenderness Ratio	Zero	$\sum a_{ij}$	$\sum \lambda_i$	λ_{max}	λ_{min}	Condition Number	Displacement
9	1.66	0.33e-6	0.23e-5	3.21e10	4.66e09	1.86e06	2.54e03	-0.0625
0.9	16.6	0.94e-6	0.12e-4	9.12e10	2.06e10	1.60e04	1.29e06	-0.0614
0.09	166.6	0.93e-5	0.97e-4	8.89e11	2.06e11	1.65e01	1.29e10	-0.0597
0.009	1666.6	0.93e-4	0.17e-2	8.88e12	2.06e12	1.56e-2	1.32e14	*

APPENDIX

ELEM03 SUBPROGRAM LISTING

The listing of subprogram ELEM03 is provided below. It includes four subroutines: NIO3, DO3, BO3 and BTDB respectively.

```
      SUBROUTINE ELEM03(VCORE,VPRNE,VPREE,VDLE,VKE,VFE)
C***********************************************************************
C    32 NODES HEXAHEDRON ELEMENT FOR 3 DIMENSIONAL ELASTICITY
C       EVALUATE ELEMENT INFORMATIONS ACCORDING TO ICODE VALUE
C          ICODE=1 ELEMENT PARAMETERS
C          ICODE=2 INTERPOLATION FUNCTIONS AND GAUSS COEFFICIENTS
C          ICODE=3 STIFFNESS MATRIX
C          ICODE=4 .
C          ICODE=5 MASS MATRIX
C          ICODE=6 RESIDUALS
C          ICODE=7 SECOND NUMBER
C          ICODE=8 EVALUATE AND PRINT STRESSES
C     ELEMENT PROPERTIES
C          VPREE(1) YOUNG'S MODULUS
C          VPREE(2) POISSON'S COEFFICIENT
C          VPREE(3) SPECIFIC MASS
C***********************************************************************
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /COOR/NDIM
      COMMON /ASSE/NSYM
      COMMON /RGDT/IEL,ITPE,ITPE1,IGRE,IDLE,ICE,IPRNE,IPREE,INEL,IDEG,
     1             IPG,ICODE,IDLE0,INEL0,IPG0
      COMMON /ES/M,MR,MP
      DIMENSION VCORE(1),VPRNE(1),VPREE(1),VDLE(1),VKE(1),VFE(1)
C
C------ CHARACTERISTIC DIMENSIONS OF THE ELEMENT
C
C     DIMENSION VCPG(IPG),VKPG(NDIM*IPG),VDE1(IMATD**2)
      DIMENSION VCPG(27),VKPG(81),VDE1(36)
C     DIMENSION VBE(IMATD*IDLE),VDE(IMATD**2),VJ(NDIM**2),VJ1(NDIM**2)
      DIMENSION VBE(576),VDE(36),VJ(9),VJ1(9)
C     DIMENSION VNIX(INEL*NDIM),VNI((1+NDIM)*INEL*IPG),IPGKED(NDIM)
      DIMENSION VNIX(96),VNI(3456),IPGKED(3)
C
C------- DIMENSION OF MATRIX D, NUMBER OF G.P.
C
      DATA IMATD/6/,IPGKED/3,3,3/
      DATA ZERO/0.0D0/,DEUX/2.D0/,X05/0.5D0/,RADN/.572957795130823D2/
      DATA EPS/1.D-6/
      DATA NNNNNN/0/
C
C--------- CHOOSE FUNCTION TO BE EXECUTED
C
      GOTO (100,200,300,400,500,600,700,800),ICODE
 100  IDLE0=96
      INEL0=32
      IPG0=27
      RETURN
C***********************************************************************
C          EVALUATE COORDINATES,WEIGHTS,FUNCTIONS N AND THEIR
C          DERIVATIVES AT G.P.
C***********************************************************************
 200  CALL GAUSS(IPGKED,NDIM,VKPG,VCPG,IPG)
      IF(M.LT.2) GOTO 220
```

```
      WRITE(MP,2000) IPG
 2000 FORMAT(/I5,' GAUSS POINTS'/10X,'VCPG',25X,'VKPG')
      I0=1
      DO 210 IG=1,IPG
      I1=I0+NDIM-1
      WRITE(MP,2010) VCPG(IG),(VKPG(I),I=I0,I1)
 210  I0=I0+NDIM
 2010 FORMAT(1X,F13.9,5X,3F13.9)
 220  CALL NI03(VKPG,VNI)
      IF(M.LT.2) RETURN
      I1=4*INEL*IPG
      WRITE(MP,2020) (VNI(I),I=1,I1)
 2020 FORMAT(/1X,'FUNCTION AND DERIVATIVES'/(1X,8E12.5))
      RETURN
C**********************************************************************
C               EVALUATE ELEMENT STIFFNESS MATRIX
C**********************************************************************
C---------- INITIALIZE VKE
 300  NINI=4656
      IF(NSYM.NE.0) NINI=9216
      DO 310 I=1,NINI
 310  VKE(I)=ZERO
C---------- FORM MATRIX D
      CALL D03(VPREE,VDE)
C---------- LOOP OVER THE G.P.
      I1=1+INEL
      DO 330 IG=1,IPG
C------ EVALUATE THE JACOBIAN,ITS INVERSE AND ITS DETERMINANT
      CALL JACOB (VNI(I1),VCORE,NDIM,INEL,VJ,VJ1,DETJ)
C---------- PERFORM  D*COEF
      C=VCPG(IG)*DETJ
      DO 320 I=1,36
 320  VDE1(I)=VDE(I)*C
C---------- PERFORM MATRIX B
      CALL DNIDX (VNI(I1),VJ1,NDIM,INEL,VNIX)
      CALL B03(VNIX,INEL,VBE)
      CALL BTDB (VKE,VBE,VDE1,IDLE,IMATD,NSYM)
 330  I1=I1+4*INEL
      RETURN
 400  CONTINUE
      RETURN
C**********************************************************************
C           EVALUATE THE MASS MATRIX
C**********************************************************************
500   NINI=4656
      IF(NSYM.NE.0) NINI=9216
      DO 510 I=1,NINI
 510  VKE(I)=ZERO
C
C           LOOP OVER THE G.P.
C
      IDIM1=NDIM-1
      IDECL=(NDIM+1)*INEL
      I1=1+INEL
      I2=0
      DO 550 IG=1,IPG
      CALL JACOB(VNI(I1),VCORE,NDIM,INEL,VJ,VJ1,DETJ)
      LL=3
      D=VCPG(IG)*DETJ*VPREE(LL)
C
C------- ACCUMULATE MASS TERMS
C
      IDL=0
      DO 540 J=1,INEL
      JJ=I2+J
      J0=1+IDL*(IDL+1)/2
      DO 530 I=1,J
      II=I2+I
      C=VNI(II)*VNI(JJ)*D
```

```
      VKE(J0)=VKE(J0)+C
      J1=J0+IDL+2
      DO 520 II=1,IDIM1
      VKE(J1)=VKE(J1)+C
 520  J1=J1+IDL+3
 530  J0=J0+NDIM
 540  IDL=IDL+NDIM
      I1=I1+IDECL
 550  I2=I2+IDECL
      RETURN
C*****************************************************************
C            EVALUATE THE ELEMENT RESIDUAL
C*****************************************************************
C-------- FORM MATRIX D
C
 600  CALL D03(VPREE,VDE)
C
C-------- INITIALIZE THE RESIDUAL VECTOR
C
      DO 610 ID=1,IDLE
 610  VFE(ID)=ZERO
C
C-------- LOOP OVER THE G.P.
C
      I1=1+INEL
      DO 640 IG=1,IPG
C-------- EVALUATE THE JACOBIAN
      CALL JACOB(VNI(I1),VCORE,NDIM,INEL,VJ,VJ1,DETJ)
C---------- EVALUATE FUNCTIONS D(NI)/D(X)
      CALL DNIDX(VNI(I1),VJ1,NDIM,INEL,VNIX)
C
C---------- EVALUATE STRAINS AND STRESSES
C
      EPSX=ZERO
      EPSY=ZERO
      EPSZ=ZERO
      GAMXY=ZERO
      GAMYZ=ZERO
      GAMZX=ZERO
      ID=1
      DO 620 IN=1,INEL
      UN=VDLE(ID)
      VN=VDLE(ID+1)
      WN=VDLE(ID+2)
      C1=VNIX(IN)
      IN1=IN+INEL
      C2=VNIX(IN1)
      IN2=IN1+INEL
      C3=VNIX(IN2)
      EPSX=EPSX+C1*UN
      EPSY=EPSY+C2*VN
      EPSZ=EPSZ+C3*WN
      GAMXY=GAMXY+C1*VN+C2*UN
      GAMYZ=GAMYZ+C2*WN+C3*VN
      GAMZX=GAMZX+C1*WN+C3*UN
 620  ID=ID+3
      C1=VCPG(IG)*DETJ
      C2=VDE(2)*C1
      C3=VDE(22)*C1
      C1=VDE(1)*C1
      SIGX=C1*EPSX+C2*EPSY+C2*EPSZ
      SIGY=C2*EPSX+C1*EPSY+C2*EPSZ
      SIGZ=C2*EPSX+C2*EPSY+C1*EPSZ
      TAUXY=C3*GAMXY
      TAUYZ=C3*GAMYZ
      TAUZX=C3*GAMZX
C
C-------- FORM THE RESIDUAL
C
```

```
      ID=1
      DO 630 IN=1,INEL
      C1=VNIX(IN)
      IN1=IN+INEL
      C2=VNIX(IN1)
      IN2=IN1+INEL
      C3=VNIX(IN2)
      VFE(ID)=VFE(ID)+C1*SIGX+C2*TAUXY+C3*TAUZX
      VFE(ID+1)=VFE(ID+1)+C2*SIGY+C1*TAUXY+C3*TAUYZ
      VFE(ID+2)=VFE(ID+2)+C3*SIGZ+C2*TAUYZ+C1*TAUZX
 630  ID=ID+3
 640  I1=I1+4*INEL
      RETURN
C*********************************************************************
C           EVALUATE BODY FORCES,FX,FY,FZ PER UNIT VOLUME
C                (FOR GRAVITY FX=0,FY=0,FZ=-VPREE(3)
C*********************************************************************
 700  FX=ZERO
      FY=ZERO
      LL=3
      FZ=-VPREE(LL)
      DO 710 I=1,96
 710  VFE(I)=ZERO
      I1=1
      IDECL=(NDIM+1)*INEL
      DO 730 IG=1,IPG
      CALL JACOB(VNI(I1+INEL),VCORE,NDIM,INEL,VJ,VJ1,DETJ)
      DX=VCPG(IG)*DETJ
      DY=DX*FY
      DZ=DX*FZ
      DX=DX*FX
      I2=I1
      I3=1
      DO 720 IN=1,INEL
      VFE(I3)=VFE(I3)+DX*VNI(I2)
      VFE(I3+1)=VFE(I3+1)+DY*VNI(I2)
      VFE(I3+2)=VFE(I3+2)+DZ*VNI(I2)
      I2=I2+1
 720  I3=I3+3
 730  I1=I1+IDECL
      RETURN
C*******************************************************************
C          EVALUATE AND PRINT STRESS AT G.P.
C*******************************************************************
 800  WRITE(MP,2080) IEL
 2080 FORMAT(//' STRESSES IN ELEMENT',I5/'   P.G.',6X,'X',11X,'Y',11X,
     1'Z',9X,'SIGX',8X,'SIGY',8X,'SIGZ',7X,'TAUXY',7X,'TAUYZ',7X,
     2'TAUZX'/)
C
C-------- FORM THE MATRIX D
C
      CALL D03(VPREE,VDE)
C
C-------- LOOP OVER THE G.P.
C
      I1=1+INEL
      I2=0
      DO 820 IG=1,IPG
C-------- EVALUATE THE JACOBIAN
      CALL JACOB(VNI(I1),VCORE,NDIM,INEL,VJ,VJ1,DETJ)
C-------- EVALUATE FUNCTIONS D(NI)/D(X)
      CALL DNIDX(VNI(I1),VJ1,NDIM,INEL,VNIX)
C
C-------- COMPUTE STRAINS AND COORDINATE AT G.P.
C
      EPSX=ZERO
      EPSY=ZERO
      EPSZ=ZERO
      GAMXY=ZERO
```

```
      GAMYZ=ZERO
      GAMZX=ZERO
      X=ZERO
      Y=ZERO
      Z=ZERO
      ID=1
      DO 810 IN=1,INEL
      UN=VDLE(ID)
      VN=VDLE(ID+1)
      WN=VDLE(ID+2)
      XN=VCORE(ID)
      YN=VCORE(ID+1)
      ZN=VCORE(ID+2)
      C1=VNIX(IN)
      IN1=IN+INEL
      C2=VNIX(IN1)
      IN2=IN1+INEL
      C3=VNIX(IN2)
      IN1=IN+I2
      C4=VNI(IN1)
      EPSX=EPSX+C1*UN
      EPSY=EPSY+C2*VN
      EPSZ=EPSZ+C3*WN
      GAMXY=GAMXY+C1*VN+C2*UN
      GAMYZ=GAMYZ+C2*WN+C3*VN
      GAMZX=GAMZX+C1*WN+C3*UN
      X=X+C4*XN
      Y=Y+C4*YN
      Z=Z+C4*ZN
  810 ID=ID+3
C
C---------- COMPUTE THE STRESSES
C
      SIGX=VDE(1)*EPSX+VDE(2)*EPSY+VDE(2)*EPSZ
      SIGY=VDE(2)*EPSX+VDE(1)*EPSY+VDE(2)*EPSZ
      SIGZ=VDE(2)*EPSX+VDE(2)*EPSY+VDE(1)*EPSZ
      TAUXY=VDE(22)*GAMXY
      TAUYZ=VDE(22)*GAMYZ
      TAUZX=VDE(22)*GAMZX
      WRITE(MP,2090) IG,X,Y,Z,SIGX,SIGY,SIGZ,TAUXY,TAUYZ,TAUZX
 2090 FORMAT(1X,I5,9E12.5)
      I2=I2+4*INEL
  820 I1=I1+4*INEL
      RETURN
      END
      SUBROUTINE NIO3(VKPG,VNI)
C***********************************************************************
C     TO EVALUATE THE SHAPE FUNCTIONS N  AND  THEIR DERIVATIVES W.R.T.
C     KSI,ETA,DZETA
C             INPUT
C                VKPG(NN)= COORDINATES OF POINTS IN KSI,ETA,DZETA
C             OUTPUT
C                VNI = THE SHAPE FUNCTION  N  AND
C       THE DERIVATIVE OF SHAPE FUNCTION W.R.T.  KSI,ETA,DZETA
C***********************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION VNI(3456),CORRF(32,3),VKPG(81),VN(32,3),RN(32)
      DATA ((CORRF(I,J),J=1,3),I=1,32)
     *           /-1.D0,-1.D0,-1.D0,-.333333333333333,-1.D0,-1.D0,0.3333
     *33333333333,-1.D0,-1.D0,1.D0,-1.D0,-1.D0,1.D0,-.333333333333333,
     *-1.D0,1.D0,.333333333333333,-1.D0,1.D0,1.D0,-1.D0,.333333333333333
     *,1.D0,-1.D0,-.333333333333333,1.D0,-1.D0,-1.D0,1.D0,-1.D0,-1.D0,
     *0.333333333333333,-1.D0,-1.D0,-.333333333333333,-1.D0,-1.D0,-1.D0,
     *-.333333333333333,1.D0,-1.D0,-.333333333333333,1.D0,1.D0,-.3333333
     *33333333,-1.D0,1.D0,-.333333333333333,-1.D0,-1.D0,.333333333333333
     *,1.D0,-1.D0,.333333333333333,1.D0,1.D0,.333333333333333,-1.D0,1.D0
     *,.333333333333333,-1.D0,-1.D0,1.D0,-.333333333333333,-1.D0,1.D0,
     *.333333333333333,-1.D0,1.D0,1.D0,-1.D0,1.D0,1.D0,-.333333333333333
     *,1.D0,1.D0,.333333333333333,1.D0,1.D0,1.D0,1.D0,.333333333333333,
```

```
     *1.D0,1.D0,-.333333333333333,1.D0,1.D0,-1.D0,1.D0,1.D0,-1.D0,.3333
     *33333333333,1.D0,-1.D0,-.333333333333333,1.D0/
C     DO 1 I=1,32
C1    WRITE(*,99) I,(CORRF(I,J),J=1,3)
C99   FORMAT (2X,I2,3(2X,E12.6))
      JJ=1
      DO 10 NN=1,81,3
      X=VKPG(NN)
      Y=VKPG(NN+1)
      Z=VKPG(NN+2)
C     PRINT*,X,Y,Z
C--------- AT CONNER
C--------- NODE # : 1,4,7,10,21,24,27,30
C
      DO 100 I=1,2
      II=20*(I-1)+1
      IT=II+9
      DO 100 J=II,IT,3
      X1=CORRF(J,1)
      Y1=CORRF(J,2)
      Z1=CORRF(J,3)
      CF1=9.D0/64.D0
      CF2=19.D0/9.D0
      RN(J)=CF1*(1.D0+X*X1)*(1.D0+Y*Y1)*(1.D0+Z*Z1)*(-CF2+X*X+Y*Y+Z*Z)
      VN(J,1)=CF1*(1.D0+Y*Y1)*(1.D0+Z*Z1)*(X1*(-CF2+3.D0*X*X+Y*Y+Z*Z)
     *        +2.D0*X)
      VN(J,2)=CF1*(1.D0+X*X1)*(1.D0+Z*Z1)*(Y1*(-CF2+X*X+3.D0*Y*Y+Z*Z)
     *        +2.D0*Y)
 100  VN(J,3)=CF1*(1.D0+X*X1)*(1.D0+Y*Y1)*(Z1*(-CF2+X*X+Y*Y+3.D0*Z*Z)
     *        +2.D0*Z)
C
C------------ AT MIDSIDE
C------------ NODE # : 2,3,8,9,22,23,28,29
C
      DO 200 K=1,2
      IK=20*(K-1)
      DO 200 I=1,2
      II=6*(I-1)+2+IK
      IT=II+1
      DO 200 J=II,IT
      X1=CORRF(J,1)
      Y1=CORRF(J,2)
      Z1=CORRF(J,3)
      CF1=81.D0/64.D0
      CF2=1.D0/9.D0
      RN(J)=CF1*(1.D0-X*X)*(CF2+X*X1)*(1.D0+Y*Y1)*(1.D0+Z*Z1)
      VN(J,1)=CF1*(1.D0+Y*Y1)*(1.D0+Z*Z1)*(X1-2.D0*X*CF2-3.D0*X*X*X1)
      VN(J,2)=CF1*Y1*(1.D0-X*X)*(CF2+X*X1)*(1.D0+Z*Z1)
 200  VN(J,3)=CF1*Z1*(1.D0-X*X)*(CF2+X*X1)*(1.D0+Y*Y1)
C
C------------- AT MIDSIDE
C------------- NODE # : 5,6,11,12,25,26,31,32
C
      DO 300 K=1,2
      IK=20*(K-1)
      DO 300 I=1,2
      II=6*(I-1)+5+IK
      IT=II+1
      DO 300 J=II,IT
      X1=CORRF(J,1)
      Y1=CORRF(J,2)
      Z1=CORRF(J,3)
      RN(J)=CF1*(1.D0+X*X1)*(1.D0-Y*Y)*(CF2+Y*Y1)*(1.D0+Z*Z1)
      VN(J,1)=CF1*X1*(1.D0-Y*Y)*(CF2+Y*Y1)*(1.D0+Z*Z1)
      VN(J,2)=CF1*(1.D0+X*X1)*(1.D0+Z*Z1)*(Y1-2.D0*Y*CF2-3.D0*Y*Y*Y1)
 300  VN(J,3)=CF1*Z1*(1.D0+X*X1)*(1.D0-Y*Y)*(CF2+Y*Y1)
C
C-------------- AT MIDSIDE
C-------------- NODE # : 13,14,15,16,17,18,19,20
```

```
C
      DO 400 J=13,20
      X1=CORRF(J,1)
      Y1=CORRF(J,2)
      Z1=CORRF(J,3)
      RN(J)=CF1*(1.D0+X*X1)*(1.D0+Y*Y1)*(1.D0-Z*Z)*(CF2+Z*Z1)
      VN(J,1)=CF1*X1*(1.D0+Y*Y1)*(1.D0-Z*Z)*(CF2+Z*Z1)
      VN(J,2)=CF1*Y1*(1.D0+X*X1)*(1.D0-Z*Z)*(CF2+Z*Z1)
 400  VN(J,3)=CF1*(1.D0+X*X1)*(1.D0+Y*Y1)*(Z1-2.D0*Z*CF2-3.D0*Z*Z*Z1)
      DO 410 L=1,32
      VNI(JJ)=RN(L)
 410  JJ=JJ+1
      DO 420 K=1,3
      DO 430 I=1,32
      VNI(JJ)=VN(I,K)
 430  JJ=JJ+1
 420  CONTINUE
 10   CONTINUE
      RETURN
      END
      SUBROUTINE DO3(VPREE,VDE)
C*********************************************************************
C     TO FORM MATRIX  D  ( 3 DIMESIONAL ELASTICITY)
C           INPUT
C                 VPREE = ELEMENT PROPERTY
C                               VPREE(1) = YOUNG'S MODULUS
C                               VPREE(2) = POISSON'S RATIO
C           OUTPUT
C                 VDE = MATRIX  D
C*********************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION VPREE(1),VDE(36)
      DATA ZERO/0.0D0/,UN/1.0D0/,DEUX/2.0D0/
      E=VPREE(1)
      LL=2
      X=VPREE(LL)
      C1=E*(UN-X)/((UN+X)*(UN-DEUX*X))
      C2=C1*X/(UN-X)
      C3=C1*(UN-DEUX*X)/(DEUX*(UN-X))
      DO 10 J=1,36
 10   VDE(J)=ZERO
      VDE(1)=C1
      VDE(2)=C2
      VDE(3)=C2
      VDE(7)=C2
      VDE(8)=C1
      VDE(9)=C2
      VDE(13)=C2
      VDE(14)=C2
      VDE(15)=C1
      VDE(22)=C3
      VDE(29)=C3
      VDE(36)=C3
      RETURN
      END
      SUBROUTINE BO3(VNIX,INEL,VBE)
C*********************************************************************
C     TO FORM MATRIX B  (3 DIMEMSIONAL ELASTICITY)
C           INPUT
C                 VNIX = DERIVATIVES OF SHAPE FUNCTION W.R.T.  X,Y,Z
C           OUTPUT
C                 VBE = MATRIX B
C*********************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION VNIX(32,1),VBE(6,1)
      DO 10 I=1,6
      DO 10 J=1,96
 10   VBE(I,J)=0.0D0
C
```

```
C                    FORMATION OF MATRIX B
C
      DO 20 I=1,32
      I1=3*I-2
      I2=I1+1
      I3=I2+1
      KK=1
      C1=VNIX(I,KK)
      C2=VNIX(I,(KK+1))
      C3=VNIX(I,(KK+2))
      VBE(1,I1)=C1
      VBE(2,I2)=C2
      VBE(3,I3)=C3
      VBE(4,I1)=C2
      VBE(4,I2)=C1
      VBE(5,I2)=C3
      VBE(5,I3)=C2
      VBE(6,I1)=C3
 20   VBE(6,I3)=C1
      RETURN
      END
      SUBROUTINE BTDB(VKE,VBE,VDE,IDLE,IMATD,NSYM)
C***********************************************************************
C     TO ADD THE PRODUCT B(T).D.B  TO  THE ELEMENT MATRIX K
C             INPUT
C                    VBE = MATRIX B
C                    VDE = MATRIX D
C             OUTPUT
C                    VKE = ELEMENT MATRIX  K
C***********************************************************************
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION VKE(1),VBE(IMATD,1),VDE(IMATD,1),T(576)
      DATA ZERO/0.0D0/
      IJ=1
      IMAX=IDLE
      DO 40 J=1,IDLE
      DO 20 I1=1,IMATD
      C=ZERO
      DO 10 J1=1,IMATD
 10   C=C+VDE(I1,J1)*VBE(J1,J)
 20   T(I1)=C
      IF(NSYM.EQ.0) IMAX=J
      DO 40 I=1,IMAX
      C=ZERO
      DO 30 J1=1,IMATD
 30   C=C+VBE(J1,I)*T(J1)
      VKE(IJ)=VKE(IJ)+C
 40   IJ=IJ+1
      RETURN
      END
```

LIST OF REFERENCES

1. Dhatt, Gouri and Touzot, Gilbert, *The Finite Element Method Displayed*, John Wiley & Sons, 1984.

2. Bathe, Klaus and Wilson, Edward, *Numerical Methods In Finite Element Analysis*, Prentice Hall, 1976.

3. Budynas, Richard G., *Advanced Strength And Applied Stress Analysis*, McGraw-Hill Company, 1977.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5002	2
3.	Department Chairman, Code 69 Department Of Mechanical Engineering Naval Postgraduate School Monterey, California 93943-5000	1
4.	Professor Gilles Cantin, Code 69Ci Department Of Mechanical Engineering Naval Postgraduate School Monterey, California 93943-5000	4
5.	Library Royal Thai Naval Academy Parknam, Samutplakan, Thailand	1
6.	LT Anan Sukanceyouth Nor-96, Naval Fleet Village Sattahip, Chonburi, Thailand	3

END

3-87

DTIC